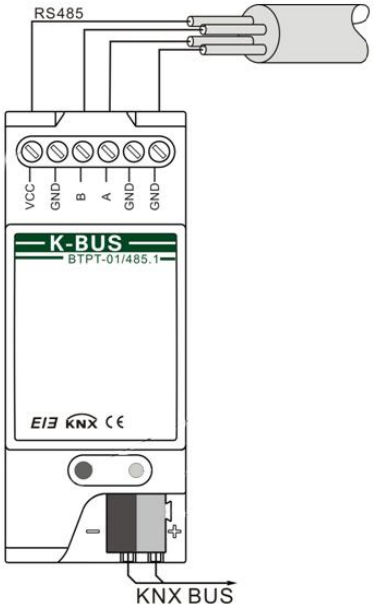
KNX/EIB RS485 Converter

User manual-Ver.1

BTPT-01/485.1



Contents

1. Summary ---- **3**

2. Technical Properties & Dimension and Circuit Diagram ---- **4**

2.1 Technical data ---- 4

2.2 Dimension diagram ---- 5

2.3 Circuit diagram ---- 5

3. Project design and programming ---- **6**

3.1 Overview of the functions ---- 6

3.2 Parameter setting description in the ETS ---- 7

4. Communication Protocol ---- **9**

4.1 Default Communication Protocol ---- 9

4.2 Command ID ---- 10

4.3 Read Group Address ---- 10

4.4 Write Group Address ---- 12

4.5 Change the Monitor Mode ---- 14

1. Summary

KNX/EIB RS485 converter is designed to convert RS485 to KNX/EIB for an intelligent building control system. It is one-way communication, built-in 2000V ESD protection. Due to 485 bus and EIB bus follow different protocol and working principle, such as the EIB bus system is a standard building control system and uses peer-to-peer control mode, while the 485 bus system uses the traditional master-slave control mode, therefore via a converter to achieve their communication and networking. Through the KNX/EIB RS485 converter to connect the 485 bus system with the EIB bus system into a whole. The device on the 485 bus sends a telegram to converter according to the predefined protocol, and then the converter will the telegram convert to KNX/EIB telegram that can be sent on the EIB bus, so that the 485 device can control the KNX device on the EIB bus.

The functions of the KNX/EIB RS485 are summarized as follows:

- Open communication protocols for the RS485 interface
- To receive and send communication telegram in accordance with preformatted
- For converting the telegram from the 485 bus to the KNX/EIB telegram, and sending to the KNX/EIB system, to control the KNX devices
- ACK report and executed result report
- Support bus monitor and filter functions (the filter table can be modified arbitrarily. Max. Number of group address is 254 in the table.)
- The baud rate of the convertor can be programmed
- To read and write any data type defined in the KNX standard (Max. 14 byte)

This manual provides detailed technical information about the KNX/EIB RS485 Converter for users as well as assembly and programming details, and explains how to use the converter by the application examples.

The KNX/EIB RS485 Converter is a modular installation device. It can be installed in the distribution board on 35 mm mounting rails according to EN 60 715. The device adopts screw terminal to achieve the 485 bus connection, using EIB bus connection terminals to connect to the KNX/EIB system, and no need an extra voltage supply.

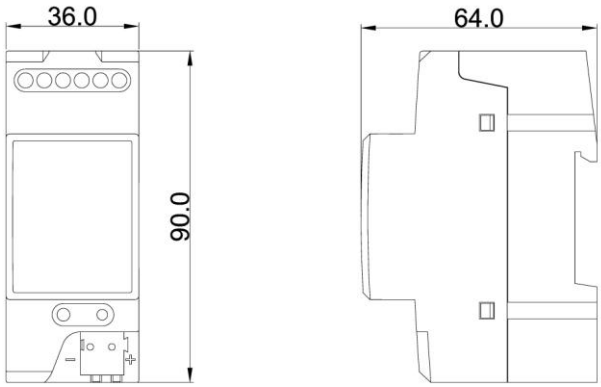
The KNX/EIB RS485 Converter is able to use the Engineering Tool Software ETS (ETS v1.3 or later) with a VD2/VD3 file to allocate the physical address and set the parameters, but a VD3 file imported is required for EST3 software.

2. Technical Properties & Dimension and Circuit Diagram

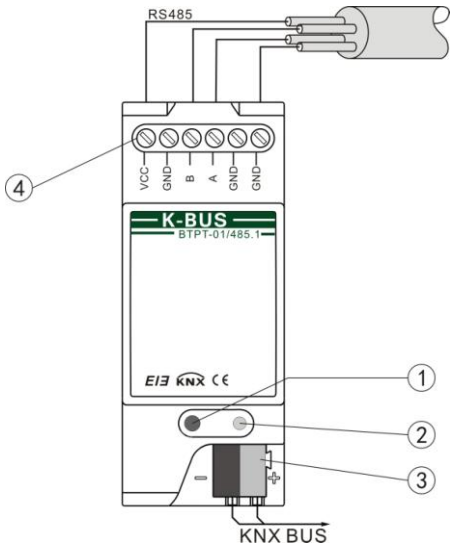
2.1 Technical data

Power supply	Operating voltage	21-30V DC，via the EIB bus
	Current input	12mA
	Power consumption via EIB	<360mW
Outputs	Number of outputs	1channel
	VCC（DC）	9V<VCC<40V(recommend DC12V/24V)
	power consumption	42mW(12V),75mW(24V)
	Max. power consumption	120mW(12V),144mW(24V)
Connections	EIB / KNX	Bus connecting terminal (black/red)
	Load circuits	1 screw terminal for VCC, 2 screw terminals for A and B bus, 3 screw terminals for GND
	Wire range	Single-core 0.2—6.0mm^2
		Multi-core 0.2—4mm^2
Operating and display elements	Red LED and push button	For assignment of the physical address
	Green LED flashing	Indicate the device running normally(OK)
Temperature range	Operation	–5 ℃ ... + 45 ℃
	Storage	–25 ℃ ... + 55 ℃
	Transport	– 25 ℃ ... + 70 ℃
Ambient condition	Humidity	<93%, except dewing
Design	Modular installation device，on 35 mm Din rail	
Interface protection	Built-in 2000V ESD protection	
Baud rate	4800~115200 bps	
Transmission range	< 1 KM	
Transmitting media	Twisted pair or shielded wire	
Working mode	Asynchronous half-duplex differential transmission	
Dimensions	90×36×64mm（H×W×D）	
Weight	0.1KG	

2.2 Dimension diagram



2.3 Circuit diagram



① Programming button

② Programming LED, red LED for assigning the physical address, green LED for indicating the application layer works normally

③ KNX / EIB bus connection terminal

④ RS485 bus connection terminal

3. Project design and programming

3.1 Overview of the functions

- **Open communication protocols**

The communication protocols for RS485 interface is open, anyone can follow the text of the protocols to achieve communication between devices on different bus. So you can easily connect the KNX/EIB bus system with other 485 bus system into a whole, but also to ensure the reliability, commonality and scalability of the system internal real-time communication, and thus really to realize openness of the system.

- **For receiving and sending communication control telegram**

The communication control telegrams are received and sent in accordance with the preformatted. The formats of the telegrams are described in the next chapter "communication protocol", i.e. command format. Each data of command represent different meaning, seeing the next chapter of the table.

- **For converting the telegram**

The telegram is received from the 485 bus by the converter, which is generated in strict accordance with the communication protocol format. When the converter receives the telegram, it will extract the EIB information from the telegram, such as the group address, length, data type and value, etc, and then the telegram is converted into the KNX/EIB telegram, and sent to the KNX/EIB system to control the KNX devices.

- **ACK report and executed result report**

When the converter receives a telegram from the 485 bus, and then the telegram is checked. If the telegram is correct, the converter will respond to the 485 bus via sending an ACK report. If the telegram is not correct, it will not respond. When the telegram is successfully transmitted to the EIB bus, the converter will send an executed result report to the 485 bus. If not successfully transmitted, there is no response.

- **Bus monitor and filter functions**

The 485 bus system can monitor the data on the EIB bus 485 via the converter. The converter also provides filter function to group address. When the filter function is set, the converter only sends the telegram which group address in the filter table to the EIB bus. The filter table can be modified arbitrarily. Max. Number of group address is 254 in the table.

- **Programming the baud rate of the convertor**

The baud rate refers to the binary data bits that are transmitted per second on the 485 bus. Different devices have different baud rate, the baud rate of the convertor can be set to the same with the devices on the 485 bus via the parameter.

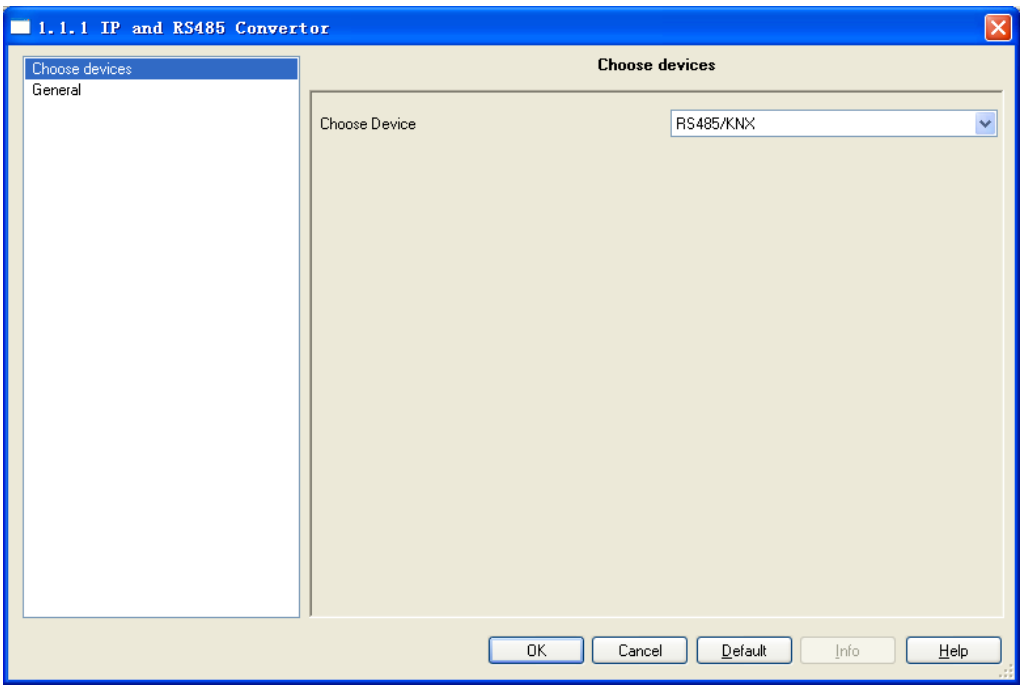
- **Reading and writing data type**

The data telegram that length is less than 14byte can be sent to the EIB bus via the converter, and the data on the EIB bus can be read.

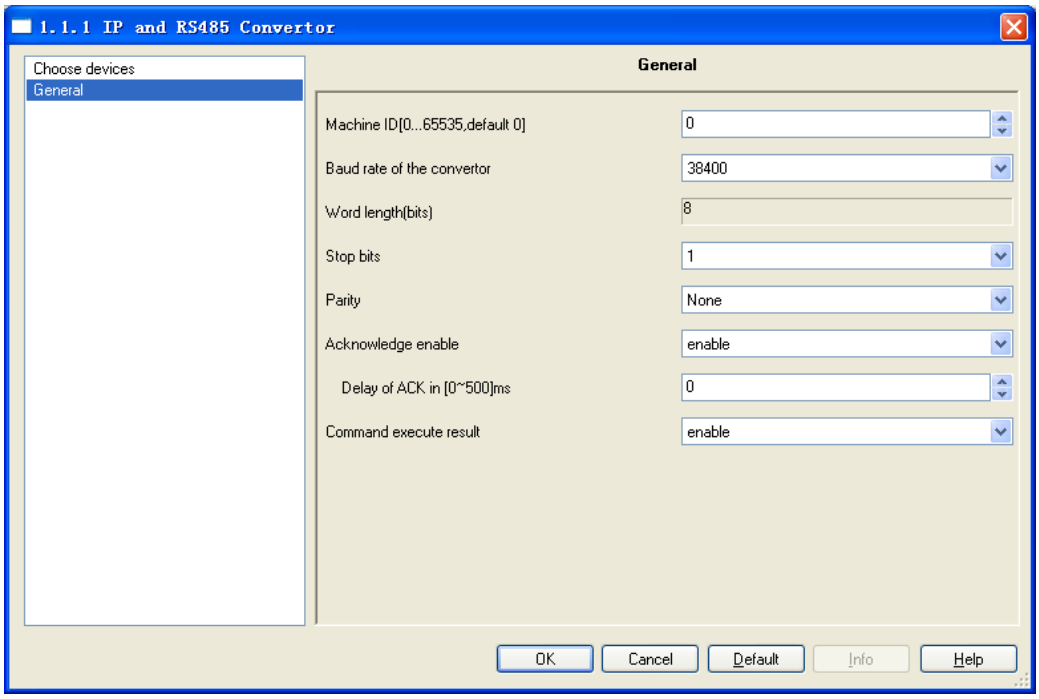
3.2 Parameter setting description in the ETS

Parameter window is shown below. Here can set some parameters about converter, such as machine ID, the baud rate, stop bits and parity, etc.

Because IP converter and RS485 converter use the same product database, so before setting RS485 parameters, you must select devices first, as follow:



Choose device



RS485 parameter window

Parameter "Machine ID [0…255， default 0]"

Here set the machine ID of the converter, the ID is a unique identifier for naming the device throughout a 485 installation. While the physical address is also a unique identifier for naming the device throughout an EIB installation. Option: 0~255

Parameter "Baud rate of the convertor"

The parameter is used to set the baud rate of the convertor. The baud rate can be set to the same with the devices on the 485 bus. Options: 4800/9600/14400/19200/38400/56000/57600/115200

Parameter "Word length (bits)"

This parameter defines word length of data transmission as 8bits. A value can be sent from 0 to 255.

Parameter "Stop bits"

This parameter defines stop bits of data transmission. Options: 0.5/1/1.5/2

Parameter "Parity"

This parameter defines parity of data transmission. Options: None/Odd/Even

Parameter "Acknowledge enable"

This parameter defines whether it sends an ACK report to the 485 bus when a 485 telegram is received successfully by the converter. Option: Enable/Disable

Parameter "Delay of ACK in [0…500] ms"

The parameter is visible in the parameter “Acknowledge enable” with “enable”, which is used to set the delay time when the converter sends an ACK report. When the delay time has elapsed, the ACK report is sent. Option: 0…500 ms

Parameter “Command execute result”

This parameter defines whether it sends a executed result report to the 485 bus when a 485 telegram is transmitted successfully to the EIB bus via the converter. Option: Enable/Disable

4. Communication Protocol

4.1 Default Communication Protocol

Baud rate of the convertor: 38400, Parity: None, Word length (bits):8, Stop bits: 1

Command packet：

```
typedef struct {
    BYTE    Head1;          // Always is 0x55=STX1.
    BYTE    Head2;          // Always is 0xAA=STX2.
    WORD    MachineID;      // terminal’s ID.
    WORD    Reserved;   // Reserved
    WORD    Command; // Command ID
    DWORD   Length;     // Input Parameter 1
    WORD    InParam;    // Input Parameter 2
    WORD    ChkSum;     // Checksum
} CMDPKT;                   // Size of command packet is 16 Bytes.
```

Command acknowledge packet：

```
typedef struct {
    BYTE    Head1;          // Always is 0x5A=STX1.
    BYTE    Head2;          // Always is 0xA5=STX2.
    WORD    MachineID;      // terminal’s ID.
    WORD    Response;  // 0x00-NAK, 0x01-ACK.
    WORD    ChkSum;     // Checksum
} ACKPKT;                   // Size of command acknowledge packet is 8 Bytes.
```

Command execution result packet：

```
typedef struct {
    BYTE    Head1;          // Always is 0xAA=STX1.
    BYTE    Head2;          // Always is 0x55=STX2.
    WORD    MachineID;      // terminal’s ID.
    WORD    Reserved;   // Reserved
```

```
    WORD    Ret;        // 1: OK 0: Error
    DWORD   OutParam;   // Output Parameter
    WORD    ChkSum;     // Checksum
} RESULTPKT;                    // Size of command execution result packet is 14 Bytes.
```

Instruction:

1、 If the command packet is arrived, the convertor sends ACK,the command should used the format of CMDPKT.

2、 After the command was executed, the convertor will report the execute result using the format of structure RESULTPKT.

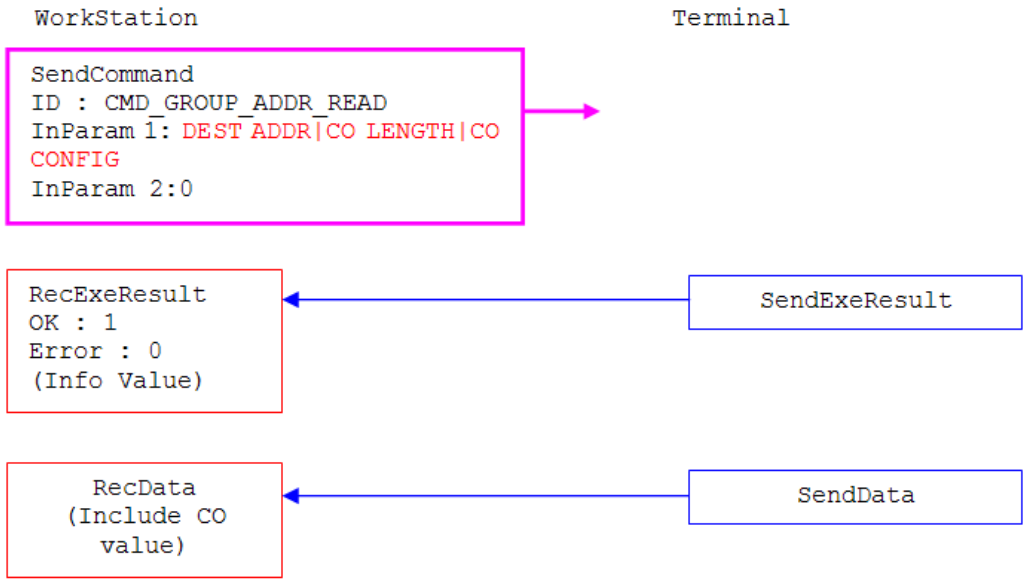
3、 When write a Group Address, more than 2 bytes Object Value should send to the Convertor after the command by Setting Data Command.

4、 When read a Group Address, insure that the Communication Object has the read/write/transmit enable. All the responds will send to external using Data Respond Command, having a header of 0Xa5-0x5A.

4.2 Command ID

```
enum // The command ID for the connection.
{
    CMD_SET_MONITOR_MODE = 0x101,
    CMD_SET_FILTER_MODE,
    CMD_SET_RANDOM_MODE,
    CMD_GROUP_ADDR_READ,
    CMD_GROUP_ADDR_WRITE
};
```

4.3 Read Group Address

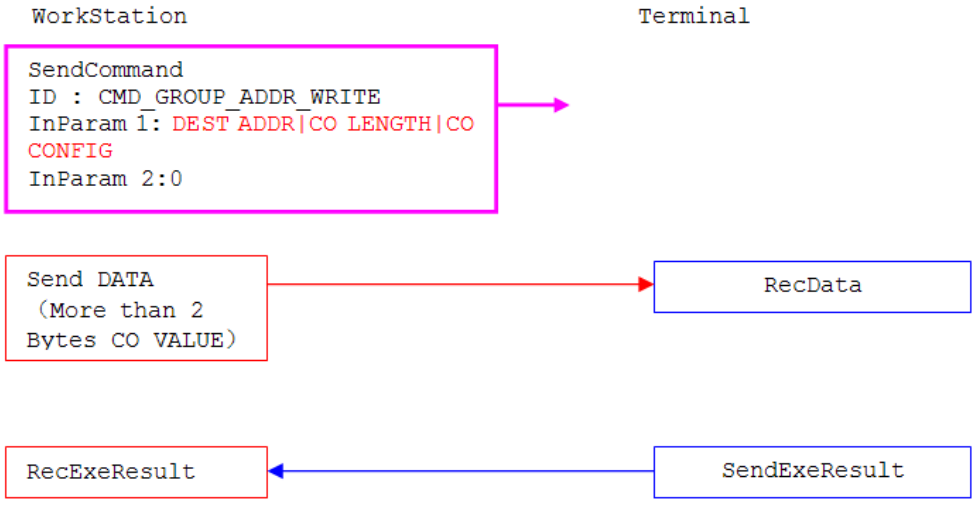


Protocol:

COMMAND(PC----Convertor)		
Byte	Value	Meaning
1	0x55	Always is 0x55
2	0xAA	Always is 0xAA
3 4	1--255, 0 = default	Machine ID
5 6	0	Reserved
7 8	CMD_GROUP_ADDR_READ	Command ID
9	nConfig(NOTE 1)(We used the lowest 2 bits as the priority, usually it equals 0x03)	Input Parameter 1
10	nLength(NOTE 2)	
11 12	DEST ADDR(NOTE 3)	
13 14	0	Input Parameter 2
15 16	ChkSum	Checksum(NOTE 4)
ACK(Convertor ---- PC)		
Byte	Value	Meaning
1	0x5A	Always is 0x5A
2	0xA5	Always is 0xA5
3 4	1--255, 0 = default	Machine ID
5 6	0x00-NAK 0x01-ACK.	Response
7 8	ChkSum	Checksum(NOTE 4)
EXERESULT (Controller ---- PC)		
Byte	Value	Meaning
1	0xAA	Always is 0xAA
2	0x55	Always is 0x55
3 4	1--255, 0 = default	Machine ID
5 6	0	Reserved
7 8	1: OK 0: Error	Result
9 10 11 12	0	Output Parameter

13 14	ChkSum	Checksum(NOTE 4)
Data respond(Convertor----PC)		
Byte	Value	Meaning
1	0xA5	Always is 0xA5
2	0x5A	Always is 0x5A
3 4	1--255, 0 = default	Machine ID
5 6	SRC ADDR(NOTE 5)	Source Address
7 8	DEST ADDR(NOTE 3)	Destination Address
9	nLength	Length of the Group Address
10 11	// Group communication Multicast APCI_VALUE_READ 0x0000 APCI_VALUE_RESP 0x0040 APCI_VALUE_WRITE 0x0080	APCI of EIB telegram
		1~14 Bytes of Value
19 20	ChkSum	Checksum(NOTE 4)

4.4 Write Group Address



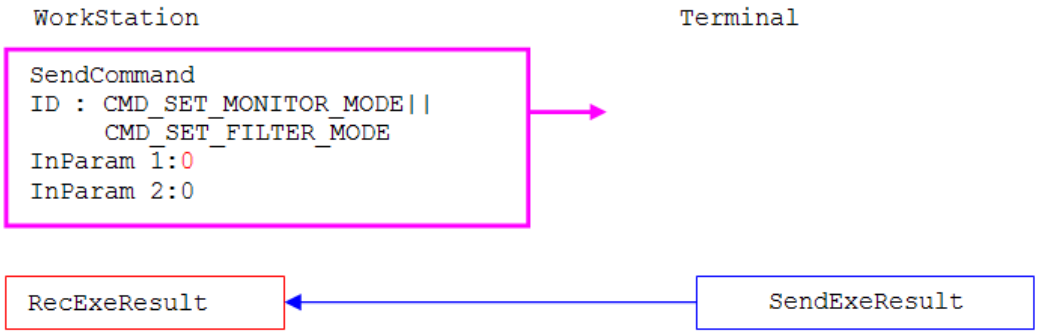
Protocol:

COMMAND(External----Convertor)		
Byte	**Value**	**Meaning**
1	0x55	Always is 0x55
2	0xAA	Always is 0xAA

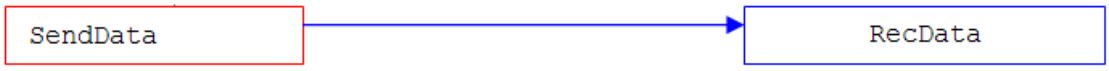
3 4	1--255, 0 = default	Machine ID
5 6	0	Reserved
7 8	CMD_GROUP_ADDR_WRITE	Command ID
9	nConfig(NOTE 1) (We used the lowest 2 bits as the priority, usually it equals 0x03)	Input Parameter 1
10	nLength(NOTE 2)	
11 12	DEST ADDR(NOTE 3)	
13 14	(nLength <= 2) : Group Value Else : 0	Input Parameter 2
15 16	ChkSum	Checksum(NOTE 4)
ACK(Convertor ---- External)		
Byte	**Value**	**Meaning**
1	0x5A	Always is 0x5A
2	0xA5	Always is 0xA5
3 4	1--255, 0 = default	Machine ID
5 6	0x00-NAK 0x01-ACK.	Response
7 8	ChkSum	Checksum(NOTE 4)
EXERESULT (Convertor ---- External)		
Byte	**Value**	**Meaning**
1	0xAA	Always is 0xAA
2	0x55	Always is 0x55
3 4	1--255, 0 = default	Machine ID
5 6	0	Reserved
7 8	1: OK 0: Error	Result
9 10 11 12	0	Output Parameter
13 14	ChkSum	Checksum(NOTE 4)
Setting Data(External ----Convertor)		
Byte	**Value**	**Meaning**

1	0x5A	Always is 0x5A
2	0xA5	Always is 0xA5
3 4	1--255, 0 = default	Machine ID
5		3~14 Bytes of Value(nLength >= 3) : Group Value
6~ 17		
18		
19 20	ChkSum	Checksum(NOTE 4)

4.5 Change the Monitor Mode



If Setting the CMD_SET_FILTER_MODE Mode，then the filter table should be send in 3 seconds after the command.



Protocol：

COMMAND(External ----Convertor)		
Byte	**Value**	**Meaning**
1	0x55	Always is 0x55
2	0xAA	Always is 0xAA
3 4	1--255, 0 = default	Machine ID
5 6	0	Reserved
7 8	CMD_SET_MONITOR_MODE || CMD_SET_FILTER_MODE	Command ID
9	CMD_SET_MONITOR_MODE：0	Input Parameter 1

10 11 12	CMD_SET_FILTER_MODE：(1～254),the length of filter table.	
13 14	0	Input Parameter 2
15 16	ChkSum	Checksum(NOTE 4)
ACK(Convertor ---- External)		
Byte	**Value**	**Meaning**
1	0x5A	Always is 0x5A
2	0xA5	Always is 0xA5
3 4	1--255, 0 = default	Machine ID
5 6	0x00-NAK 0x01-ACK.	Response
7 8	ChkSum	Checksum(NOTE 4)
EXERESULT(Convertor ---- External)		
Byte	**Value**	**Meaning**
1	0xAA	Always is 0xAA
2	0x55	Always is 0x55
3 4	1--255, 0 = default	Machine ID
5 6	0	Reserved
7 8	1: OK 0: Error	Result
9 10 11 12	0	Output Parameter
13 14	ChkSum	Checksum(NOTE 4)
Setting Filter Table(External ----Convertor)		
Byte	**Value**	**Meaning**
1	0x5A	Always is 0x5A
2	0xA5	Always is 0xA5
3 4	1--255, 0 = default	Machine ID
	High Byte of the Group Address Low Byte of the Group Address High Byte of the Group Address Low Byte of the Group Address	Max Data:254*2 Bytes(The Lower Byte of the Group Address follows the higher Byte, and they should be marked in ascending order, e.g., Group Address 00-05 should places before 00-06, also 00-05 should places

	…High Byte	before 01-05).
	…Low Byte	
	…High Byte	
	…Low Byte	
19	ChkSum	Checksum(NOTE 4)
20		

NOTE：

NOTE 1: According to KNX Standard, it has 4 types, indicates 4 Transmission Priority:

11 = low operational priority

10 = high operational priority

01 = alarm priority

00 = system priority

Usually we use LOW priority, so the nConfig will be 0x03, others are not recommended.

NOTE 2: The length of the Communication Object which was linked to the DEST ADDR according to the KNX standard, e.g. when the CO lest than 1 byte it equals 1, otherwise it equals to length + 1.That is to say, when we have a 1 bit CO, then the nLength equals 0x01, while a 1 byte CO will have the nLength equals 0x02.

NOTE 3: Destination Group Address which you want to read or write.

NOTE 4: The sum of all the data bytes before.

NOTE 5: The Physical Address of the device from whom sends out the telegram.